Plain English explanation of D0 Streaming Plan - L3 to Physical Streaming

H. Schellman

Version 1.0 June 14, 2001

Contents

1 Introduction

This is a summary of many notes and memos which can be found at www-d0.fnal.gov/~schellma/upgrade_computing

The problem we wish to solve is that of taking the 2^N, where $N > 100$ possible combinations of **L3 filters scripts** (colloquially known as 'L3 triggers') [1], which could fire in any given event and mapping them onto a small number of **Physical Streams**, collections of files or tapes which we store and use for data analysis.

We need some way of classifying events which is coarser than **L3 triggers**.It was proposed in D0 note 3313 in 1997 that this classification be done on the basis of **Objects** in the event, for example, electrons, muons and jets.

This leads to a somewhat complicated mapping between 'L3 triggers' and **Physical streams**, as one must first classify the trigger based on the objects it is sensitive to and then find which **Physical Stream**s those objects belong in. This note will describe that implementation, much of which is already present in the L3 code.

2 How L3 'triggers' are categorized

2.1 Use of Objects

There are many possible **Objects** in events, separated vertices, jet with 5 tracks etc. We have decided to group these objects into entities called StreamPrimitives, which correspond to classes of physics objects.

The first example implementation had 5 StreamPrimitives:

ELECTRON

LOWPTMU

HIPTMU

[1] I know this nomenclature is not what the L3 group uses but it's what most people understand.

MISSINGET

JET

A more realistic implementation might also add taus, B candidates etc.. We can presume that there will be around 10 **Primal Streams** in RunIIa. [2]

2.2 L3 filters

Each **L3 filter** has a small number of **Primal Streams** associated with it. For example a 'separated tau' might have primitives 'SEPVERT' and 'TAU'.

2.3 L3 filter scripts

Each **L3 filter script** 'L3 trigger' is a combination of **L3 filters** and any event has the OR of all of the **Primal Stream**s corresponding to the L3 filters which passed it.

If the script is just an AND of all of the individual filters, each L3 trigger will have a unique Stream Bitmap. This is guaranteed by the current system.

These assignments are part of the L3 filter descriptions and are stored in the trigger database so that they can be accessed later.

3 How events are categorized

3.1 Classifying Events

The classification of an event in terms of **Primal Streams** is just the OR of all of the **Primal Streams** for L3 'triggers' it passed. One can say that one is classifying based on the **Objects** in the event but there is a clear and deterministic relation with the L3 'triggers' which is constant for any given trigger list.

[2] The system has been written to allow the evolution of **Primal Stream** definitions with time. So if they are stored as bits 32 to 64 are available.

This classification is stored in the event record and is used to determine which data stream the event goes to.

If there are n **Primal Streams** we have reduced the 2^N problem to 2^n but that still leaves us with $\sim$ 1000 possible event classifications or **Stream Bitmaps**[3] , some of which will be very common and some of which will be very rare.

3.2 Mapping onto Physical Streams

We wish to group the ~1000 possible Stream Bitmaps into a small number (5-30) of actual **Physical Stream**s for output. The criteria are that the events in a **Physical Stream** are similar to each other, that no **Physical Stream** be too large or too small and that the algorithm be reconstructable from information in the trigger database.

Zhong Yu proposed a splitting algorithm which yields a decision tree resulting in an semi-optimal set of streams. The algorithm can be encoded as a set of decision lists, one for each stream and thus quite short. This does not always yield the optimal algorithm but it does provide a simple way of mapping Stream Bitmaps to **Physical Stream**s which is easy to describe and encode.

His original proposal was an automatic algorithm, which is probably necessary at some level due to the complexity of the problem, but which can be reoptimized in light of physics needs without compromising the encoding ability.[4]

The L3 code can use this algorithm to tell the DataLogger where to place an event.

[3] These 2^n combinations of **Primal Streams** were originally called Stream Bitmaps but somehow got called LogicalStreams when the L3 code was implemented. Unfortunately, LogicalStream was used to denote the combination of PhysicalStreams needed for a data analysis in early memos. Here I'm going to stick with Stream Bitmap.

[4] For example, if one wished to ensure that all W triggers go to a single stream, one could impose that the first split be on ELECTRON and that the second split on the ELECTRON branch be on MISSING ET with termination at that point. The $\overline{ELECTRON}$ and $ELECTRON + \overline{MISSINGET}$ branches could continue with the automatic algorithm.

4 Doing Physics analysis

When doing physics analysis one generally asks for a given trigger or set of triggers. One wishes to read as few data files as possible but to find all of the triggers you requested.

The procedure is as follows.

1. Determine which trigger you wish.
2. Loop over run numbers, for each run, find the appropriate trigger list in the db
3. For each trigger list, the**Primal Stream**s associated with that trigger are known.
4. From the trigger list, extract the algorithm parameters for the streaming.
5. Run the algorithm on the**Primal Stream**s for your trigger to find the subset of **Physical Stream**s which contain that trigger. One could also use a plain lookup table, where each **Physical Stream** has a list of the Stream Bitmaps which went to that **Physical Stream** for that Trigger list.

The general user would not do this. D0 communicates with the Oracle Database via a DbServer, which, because it is written in python and C++ can perform more complicated actions than a raw Oracle DB. The user should be able to say

"Give me the **Physical Stream**s which contain Trigger X for Run B"

and the DbServer can translate the trigger into**Primal Stream**s and then compare with the **Physical Stream** specifications.

5 Conclusion

Simple algorithms exist for translating between the physical location of data on disk and tape and the trigger content of those data. They are non-trivial but can be implemented using the existing D0 database and L3 codes.

5.1 Status

The L3 algorithms to determine the **Stream Bitmaps** exist and are in CVS. The Primal Stream assignements are part of the trigger list.

The standalone algorithms for mapping between triggers and Physical streams are understood but not fully implemented. They require

- A small amount of code for use in L3 to determine the Stream for each event.
- One or more database tables associated with the trigger list to store the algorithm parameters.
- A dbserver interface which allows queries mapping triggers to Stream Bitmaps and Stream Bitmaps to Physical Streams.
- Political decisions by the trigger board on the objects to be used as Primal Streams and the details of the decision tree.